Amendments to the Claims:

The following listing of the claims will replace all prior versions and listings of claims in the application.

Listing of the Claims:

1. (Currently Amended) A computer implemented method of detecting vulnerabilities in a pre-existing source code listing, stored in computer readable medium having computer executable instructions, said source code having an inherent control flow and an inherent data flow during the computer execution thereof, said computer implemented method comprising the acts of:

analyzing computer-executable variables in the source code listing in the context of at least one of the inherent control flow and inherent data flow, and creating models therefrom in which each model specifies pre-determined characteristics about each variable;

using the variable models to create models of arguments to routine calls in the source code listing; ~~and~~

using the argument models in conjunction with pre-specified criteria for the corresponding routine calls to determine whether the routine calls possess vulnerabilities as a consequence of the arguments and known routine behavior; and

generating a report that identifies the vulnerabilities.

2. (Currently Amended) The computer implemented method of claim 1 wherein the models specify the memory size of a variable.

3. (Currently Amended) The computer implemented method of claim 1 wherein the models specify the data size of a variable.

4. (Currently Amended) The computer implemented method of claim 1 wherein the models specify whether the variable is a null terminated string or not null terminated string for variables of string value type.

5. (Currently Amended) The computer implemented method of claim 1 wherein the models specify the type of memory of the variable.

6. (Currently Amended) The computer implemented method of claim 1 wherein the models specify the value of a string for variables that are of string value type.

7. (Currently Amended) The computer implemented method of claim 1 wherein the models specify the origin of the data for a variable.

8. (Currently Amended) The computer implemented method of claim 1 wherein the argument models specify characteristics of variable arguments.

9. (Currently Amended) The computer implemented method of claim 1 wherein the argument models specify characteristics of expression arguments.

10. (Currently Amended) The computer implemented method of claim 1 wherein the models are specified as lattices.

11. (Currently Amended) The computer implemented method of claim 10 wherein the lattice values ~~can~~ include at least one of a value to represent no knowledge, a value to represent inconsistent knowledge, and a value to represent a refinement of knowledge.

12. (Currently Amended) The computer implemented method of claim 11 wherein the value to represent a refinement of knowledge includes values to specify a range of specific values.

13. (Currently Amended) The computer implemented method of claim 1 wherein the pre-specified criteria for the corresponding routine includes rules about the semantic behavior of the routine.

14. (Currently Amended) The computer implemented method of claim 1 wherein the vulnerabilities are buffer overflows.

15. (Currently Amended) A computer implemented method of detecting vulnerabilities in a pre-existing source code listing stored in computer readable medium having computer executable instructions, said source code having an inherent control flow and an inherent data flow during the computer execution thereof, said computer implemented method comprising the acts of:

analyzing the source code listing in the context of at least one of the inherent control flow and inherent data flow to create models of arguments to routine calls in the source code listing, ~~and~~

using the argument models in conjunction with pre-specified criteria for the corresponding routine calls to determine whether the routine calls possess vulnerabilities as a consequence of the arguments and the routine behavior~~.~~ ; and

generating a report that identifies the vulnerabilities.

16. (Currently Amended) A computer implemented utility ~~system~~ for detecting vulnerabilities in a pre-existing source code listing stored in computer readable medium having

computer executable instructions, said source code having an inherent control flow and an inherent data flow during the computer execution thereof, comprising:

computer implemented logic for analyzing computer-executable variables in the source code listing and creating models therefrom in the context of at least one of the inherent control flow and data flow, in which each model specifies pre-determined characteristics about each variable;

computer implemented logic for using the variable models to create models of arguments to routine calls in the source code listing; and

computer implemented logic for using the argument models in conjunction with pre-specified criteria for the corresponding routine calls to determine whether the routine calls possess vulnerabilities as a consequence of the arguments and known routine behavior; and

computer implemented logic for generating a report that identifies the vulnerabilities.

17. (Currently Amended) The computer implemented utility ~~system~~ of claim ~~20~~ 16, using a data base having computer readable information about a predefined set of source code routine calls, said information specifying one ore more conditions that present a vulnerability during execution of the source code routine call, wherein the computer~~d~~ implemented logic for using the argument models in conjunction with pre-specified criteria for the corresponding routine calls to determine whether the routine calls possess vulnerabilities as a consequence of the arguments and known routine behavior includes computer implemented logic for using the ~~a~~ database to retrieve information for a corresponding routine call to check for the specified condition to see whether the routine

call presents a vulnerability. ~~specifying rules to detect vulnerabilities based on an analysis of the argument models.~~

18. (New) The computer implemented method of claim 1, using a data base having computer-readable information about a predefined set of source code routine calls, said information specifying one or more conditions that present a vulnerability during execution of the source code routine call, wherein the act of using the argument models in conjunction with pre-specified criteria for the corresponding routine calls to determine whether the routine calls possess vulnerabilities as a consequence of the arguments and known routine behavior comprises the act of using the data base to retrieve information for a corresponding routine call to check for the condition to see whether the routine call presents vulnerability.

19. (New) The computer implemented method of claim 15, using a data base having computer-readable information about a predefined set of source code routine calls, said information specifying one or more conditions that present a vulnerability during execution of the source code routine call, wherein the act of using the argument models in conjunction with pre-specified criteria for the corresponding routine calls to determine whether the routine calls possess vulnerabilities as a consequence of the arguments and the routine behavior comprises the act of using the data base to retrieve information for a corresponding routine call to check for the condition to see whether the routine call presents a vulnerability.

20. (New) The computer implemented method of claim 1 wherein the report identifies the location in the source code listing where the vulnerability occurred.

21. (New) The computer implemented method of claim 15 wherein the report identifies the location in the source code listing where the vulnerability occurred.

22. (New) The computer implemented utility of claim 16 wherein the report identifies the location in the source code listing where the vulnerability occurred.